

Estudis Universitaris de Vic
Escola Universitària Politècnica d'Osona
Adscrita a la Universitat Politècnica de Catalunya
Enginyeria Tècnica de Telecomunicació
especialitat en Sistemes de Telecomunicació

ADQUISICIÓ I DISTRIBUCIÓ DEL SENYAL
manual de pràctiques

Antoni Suriñach i Albareda
Departament d'Electrònica

ÍNDEX

Pràctica 1
SISTEMA DE CONTROL DEL NIVELL D'UN DIPÒSIT

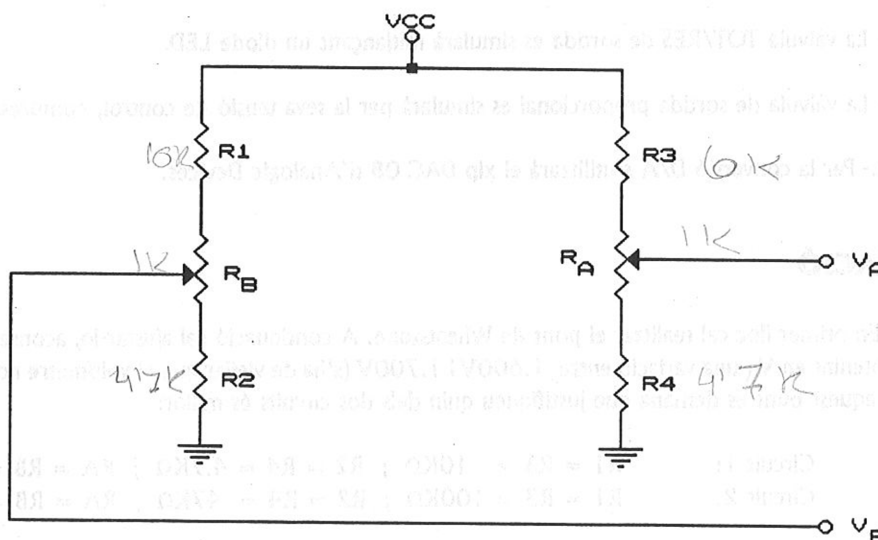
OBJECTIUS

- Estudi dels ponts de Wheatstone.
- Estudi dels amplificadors d'instrumentació.
- Estudi dels convertidors A/D i D/A.
- Estudi de la placa d'E/S digital.
- Estudi del condicionament de senyal mitjançant un P.C.
- Realització d'un circuit de mesura mitjançant un transductor muntat en un pont de Wheatstone.
- Realització d'un circuit de condicionament mitjançant un amplificador d'instrumentació.
- Realització d'un circuit de conversió A/D.
- Realització d'un sistema software que efectuï l'adquisició i el tractament de les dades subministrades pel transductor.
- Realització d'un circuit de conversió D/A.

ESPECIFICACIONS

1.- Disseny i implementació d'un sistema de mesura i control del nivell d'un dipòsit.

2.- El detector de nivell serà un transductor diferencial de pressió col.locat al fons del dipòsit, i connectat en pont de Wheatstone. Simularem el sistema transductor-pont amb el circuit de la figura següent, on el transductor es substitueix per un potenciòmetre multivolta.



3.- El nivell màxim del líquid en el dipòsit és d'1.00m, i es correspon amb una tensió de sortida del transductor V_A = 1.700V. El nivell mínim del líquid és de 0.00m, i es correspon amb una tensió de sortida del transductor V_A = 1.600V.

4.- El circuit condicionador serà un amplificador d'instrumentació realitzat amb 3 amplificadors operacionals discrets, oferint un marge de la tensió de sortida adeqüat al marge dinàmic del convertidor A/D.

5.- Per la conversió A/D s'utilitzarà el xip ADC 0804 d'Analogic Devices.

6.- Les entrades/sortides de l'ordinador es realitzaran a partir de la placa d'E/S digital, que disposa de dos xips 8255 d'Intel.

7.- El tractament digital del senyal serà:

Conversió de la tensió d'entrada en el corresponent nivell de líquid.

Filtrat i/o linialització digital dels valors d'entrada.

Control del nivell del líquid.

Activació, si s'escau, de les vàlvules de control.

Representació del sistema i actualització dels seus paràmetres característics en la pantalla de l'ordinador personal.

La rutina de control del sistema estarà basada en el següent algorisme:

L'operari podrà entrar uns valors de "Nivell Màxim", "Nivell Mínim", "Histèresi Màxima" i "Histèresi Mínima".

La rutina controlarà que el nivell de líquid estigui comprès entre els valors mínim i màxim.

En cas que el nivell sigui superior o igual al màxim, el sistema obrirà una vàlvula de sortida de líquid del tipus TOT/RES, i no la tancarà fins que el nivell hagi retornat a un valor inferior o igual al nivell màxim menys la histèresi màxima.

En cas que el nivell sigui inferior o igual al mínim, el sistema obrirà una vàlvula d'entrada de líquid del tipus proporcional, oferint un cabal d'entrada proporcional al valor que falti per arribar al nivell mínim més la histèresi mínima.

8.- La vàlvula TOT/RES de sortida es simularà mitjançant un díode LED.

9.- La vàlvula de sortida proporcional es simularà per la seva tensió de control, compresa entre 0V i +5V.

10.- Per la conversió D/A s'utilitzarà el xip DAC 08 d'Analogic Devices.

REALITZACIÓ

a) En primer lloc cal realitzar el pont de Wheatstone. A continuació cal ajustar-lo, aconseguint $V_B = 1.600$ V i obtenint en V_A una variació entre 1.600V i 1.700V (s'ha de vigilar que el voltímetre no carregui el circuit). En aquest punt es demana que justifiqueu quin dels dos circuits és millor:

Circuit 1:	R1 = R3 = 10KΩ ;	R2 = R4 = 4.7KΩ ;	RA = RB = 1KΩ
Circuit 2:	R1 = R3 = 100KΩ ;	R2 = R4 = 47KΩ ;	RA = RB = 10KΩ

si tenim en compte que la impedància d'entrada del voltímetre digital és $R_{in} = 10M\Omega$, i la impedància d'entrada d'un A.I. és de l'ordre de GΩ. Com ho farieu per ajustar cada un dels dos circuits?. Monteu el circuit 1.

b) En segon lloc cal realitzar l'amplificador d'instrumentació a partir de components discrets. Cal compensar les tensions d'offset i de mode comú i aconseguir una tensió de sortida compresa entre 0V i V_{FE} del CAD.

c) En tercer lloc cal realitzar el circuit convertidor A/D, ajustant de manera convenient l'offset i "l'span" del convertidor, per tal d'obtenir una lectura &H00 per un nivell del líquid de 0 mm, i una lectura de &HFF per un nivell de 1000 mm. En aquest punt es pot optar per dos circuits diferents:

Fer funcionar el convertidor A/D en mode "free-running", es a dir, que efectuï conversions a la velocitat del seu rellotge intern, sense intervenció de cap rutina de control. En aquest cas l'adquisició de dades per part de l'ordinador s'efectua sense cap mena de sincronisme i pot fer falta, probablement, algun tipus de filtratge o processat del senyal.

Fer funcionar el convertidor A/D de manera controlada per l'ordinador. En aquest cas el sistema es complica perquè apareix un protocol de comunicacions entre el P.C i el convertidor A/D, però el sincronisme de la transferència de dades el porta l'ordinador i, per tant, no farà falta el processat especial dels senyals.

d) En quart lloc cal realitzar la rutina de control. Aquest software tindrà diferents funcions:

- La primera funció serà adquirir les dades de pressió que subministra el transductor. En aquest apartat caldrà generar els senyals de control pel CAD en cas que s'hagi optat pel segon circuit.

- La segona funció serà convertir aquestes dades de pressió en altura corresponent de líquid. En cas que sigui necessari s'aplicarà un processat consistent en un filtre digital i/o una linialització digital per eqüació o taula.

- La tercera funció serà efectuar el control del nivell del líquid:

1- En el moment que el nivell del líquid sigui inferior o igual al valor mínim programat, la sortida del DAC serà de +5V (s'obrirà la vàlvula d'entrada). A mesura que el nivell del líquid vagi augmentant, la tensió a la sortida del DAC disminuirà proporcionalment, arribant a valer 0V quan el nivell del líquid sigui superior o igual al valor del nivell mínim més la histèresi mínima.

2- En el moment que el nivell del líquid superi o iguali el valor màxim programat, s'activarà la vàlvula de sortida de líquid, encenent-se el díode LED. Quan el nivell retorni a un valor inferior o igual al valor màxim menys la histèresi màxima, es tancarà la vàlvula de sortida, apagant-se el díode LED.

- La quarta funció serà dibuixar el sistema en la pantalla de l'ordinador i actualitzar els seus paràmetres. Una possible ampliació d'aquest apartat consisteix en la implementació d'un histograma en temps real, que vagi deixant constància dels valors que es succeeixin en el nivell del dipòsit.

- La cinquena funció serà la interfície amb l'usuari, restant alerta al teclat o a la "rata" per si l'operari vol canviar els paràmetres de control del sistema.

e) Finalment cal realitzar el circuit de control de les vàlvules d'e/s, format pel DAC08 i el díode LED. La tensió de sortida del DAC estarà compresa entre 0 i 5V.

ES VALORARÀ

1) La realització i bon funcionament de tot el circuit (amplificador d'instrumentació, convertidor A/D, software de control, convertidor D/A).

2) L'exactitud i linealitat del circuit de mesura i control.

3) Les ampliacions que s'hagin implementat satisfactòriament (sincronització del CAD, linialització digital del transductor-pont de Wheatstone, realització de l'histograma,...).

4) La presentació gràfica del programa de control i una interfície fàcil i agradable amb l'usuari.

Pràctica 2
OSCIL.LOSCOPI DIGITAL I ANALITZADOR DE FOURIER

OBJECTIUS

- Estudi d'una placa comercial d'adquisició de dades: PCL-711B, de la firma Advantech Co, Ltd.
- Estudi dels sistemes de transferència de dades entre la placa d'adquisició de dades i el P.C.
- Estudi del processat digital elemental del senyal.
- Estudi de la representació gràfica en la pantalla d'un P.C.
- Realització d'un sistema que adquireixi un senyal extern, el processi digitalment obtenint-ne els seus paràmetres fonamentals, incloent-hi l'FFT i els representi en la pantalla del P.C.

ESPECIFICACIONS

1.- Realització d'un instrument virtual format per un oscil.loscopi digital i un analitzador de Fourier elementals, a partir d'una tarja comercial d'adquisició de dades i un ordinador personal P.C.

2.- El senyal d'entrada s'obtindrà d'un generador de senyals del laboratori, amb una tensió màxima de sortida de $\pm 5V$.

3.- El sistema d'adquisició de dades consistirà en la placa PCL-711B de la casa Advantech Co., Ltd., i s'utilitzarà un dels 8 canals analògics d'entrada de que disposa.

4.- La programació de la tarja s'efectuarà en una de les dues modalitats:

a) Programació de la tarja a nivell de registres interns. Amb aquesta modalitat s'aconsegueix una freqüència de mostreig propera als 40KM/s, però és difícil transferir les dades mitjançant interrupcions (no es pot treballar en background), i és difícil sincronitzar el mostreig amb el rellotge intern.

b) Programació de la tarja mitjançant crides a funcions externes. Amb aquesta modalitat només s'aconsegueix una freqüència de mostreig inferior als 15KM/s, però és fàcil transferir les dades mitjançant interrupcions (es pot treballar en background), i també és fàcil sincronitzar el mostreig amb el rellotge intern.

La primera ampliació de la pràctica consisteix en implementar les dues modalitats de programació de la tarja. La segona ampliació consisteix en la realització d'un instrument virtual de dos canals d'entrada.

5.- El processat digital del senyal d'entrada consistirà, com a mínim, en:

a) Obtenció dels paràmetres característics d'aquest senyal: Freqüència, Període, Tensió pic a pic, Tensió màxima, Tensió mínima, Tensió promig, Tensió eficaç, etc...

b) Càlcul de l'FFT del senyal, amb l'obtenció dels valors que es dibuixaran en els eixos segons quina sigui la freqüència de mostreig. Pel càlcul de l'FFT es pot utilitzar la rutina que s'inclou en aquest manual de pràctiques, o realitzar una rutina pròpia.

En aquest apartat s'obre el camp a noves ampliacions, ja que tot o part del processat del senyal d'entrada es pot realitzar en un sol període, o bé promitjant els resultats aconseguits en varis períodes. També es pot dissenyar un sistema que adquireixi el senyal d'entrada a una freqüència de mostreig fixa, o bé que la freqüència de mostreig sigui programable per l'usuari.

6.- La visualització gràfica dels dos instruments virtuals es realitzarà simultàniament en la pantalla de l'ordinador personal, mostrant, per cada instrument, la forma d'ona i els paràmetres calculats anteriorment. Per la representació gràfica dels dos senyals es poden utilitzar les rutines incloses en el manual de pràctiques o be es poden implementar rutines pròpies.

7.- El llenguatge de programació serà el Turbo C/C++.

ES VALORARÀ

1) El compliment de les especificacions mínimes demanades.

2) L'exactitud i linealitat dels senyals obtinguts i de les dades calculades.

3) La quantitat, dificultat i grau d'implementació de les ampliacions realitzades.

4) La presentació gràfica de l'instrument virtual i la interfície fàcil i agradable amb l'usuari.

Pràctica 3

GRAVADOR/REPRODUCTOR DIGITAL DE VEU AMB UN P.C.

OBJECTIUS

- Estudi d'un sistema bàsic de gravació/reproducció de veu amb un ordinador personal, mitjançant la utilització d'una tarja comercial d'adquisició de dades: PCL-711B, de la firma Advantech Co, Ltd.
- Estudi del procés d'(amplificació-mostreig-conversió A/D-adquisició-gravació en disc dur) de la veu.
- Estudi del procés de (lectura de disc dur-mostreig de sortida-conversió D/A-filtrat-amplificació) de la veu.
- Realització d'un sistema de gravació de senyals acústics en fitxers desats en el disc dur d'un ordinador personal, i d'un sistema de reproducció d'aquests mateixos fitxers.

ESPECIFICACIONS

1.- Realització d'un sistema que adquireixi un senyal acústic mitjançant un micròfon, el processi digitalment i el desi en un fitxer al disc dur de l'ordinador personal. El sistema també podrà recuperar els senyals desats en aquests fitxers, obtenint-ne la corresponent sortida acústica per un altaveu.

2.- El senyal d'entrada s'obtindrà d'un micròfon connectat a un amplificador d'àudio que ofereixi un marge de sortida adequat al marge d'entrada de la tarja d'adquisició PCL-711B.

3.- L'adquisició del senyal s'efectuarà mitjançant la tarja PCL-711B, programant adequadament la freqüència de mostreig. Seria bo que entre la pràctica 2 i 3 s'aconseguís programar la tarja en les dues modalitats que proporciona.

4.- Una vegada adquirida i processada una dada, aquesta es desarà en un fitxer DOS al disc dur. En aquest apartat convé tenir en compte que la freqüència màxima de mostreig vindrà condicionada pel temps que trigui l'ordinador en adquirir la dada, processar-la i desar-la en el disc dur.

5.- La recuperació d'un senyal acústic desat en un fitxer s'obtindrà a partir de la conversió D/A de les dades d'aquest fitxer i del seu re-mostreig a la freqüència convenient. Una vegada obtingut el senyal analògic, aquest es podrà filtrar i amplificar. En aquest apartat hi haurà la possibilitat de practicar amb diferents freqüències de mostreig, tant en la conversió A/D com en la conversió D/A, a fi d'observar el seu efecte en el senyal reconstruït.

6.- Una possible ampliació de la pràctica consisteix en l'aplicació d'un processat digital que permeti canviar el to d'un senyal acústic (per exemple, canviar una veu masculina per una de femenina o a l'inrevés).

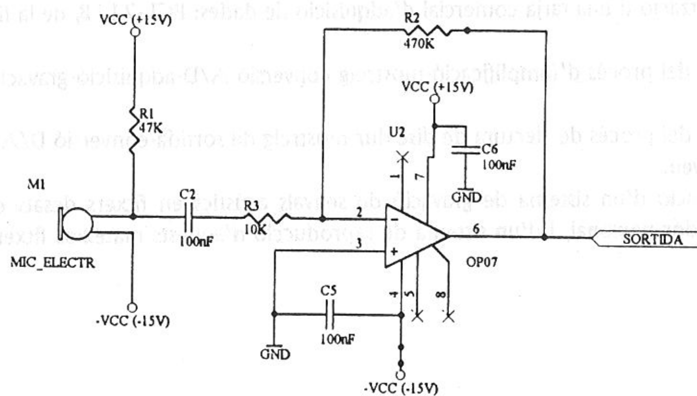
ES VALORARÀ

1) El compliment de les especificacions mínimes demanades.

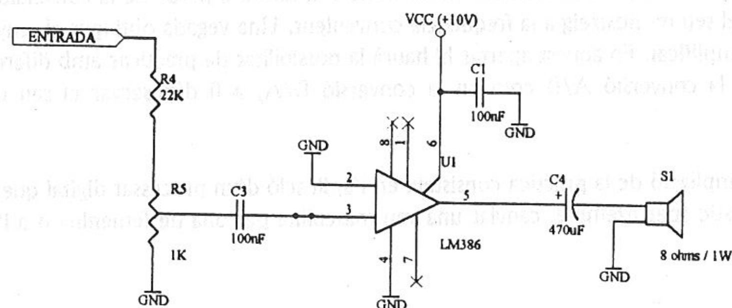
2) La quantitat, dificultat i grau d'implementació de les ampliacions realitzades.

Esquema electrònic de l'amplificador del micròfon i de l'amplificador de sortida:

Etapa d'entrada



Etapa de sortida



```
/*************************************************************/
/*                                                           */
/*              OSCIL.LOSCOPI DIGITAL I FFT                  */
/*                                                           */
/*                      Pràctica 2                           */
/*                                                           */
/*         E.U.V.  Departament d'Electrònica                 */
/*                                                           */
/*             Versió 3,   4 de març de 1.996                */
/*                                                           */
/*************************************************************/

#include <graphics.h>
#include <math.h>
#include <stdio.h>
#include <dos.h>

#define N_MOSTRES 512        /* Nombre de mostres (múltiple de 2^n per l'FFT) */
#define x1 50                /* Posició X display oscil.loscopi (mínim 40) */
#define y1 20                /* Posició Y display oscil.loscopi */
#define x2 50                /* Posició X display FFT (mínim 40) */
#define y2 200               /* Posició Y display FFT */

int taula[N_MOSTRES+2];      /* Taula original amb els valors de l'ona que s'han de dibuixar */
int taulab[N_MOSTRES+2];     /* Còpia de la taula que s'utilitza per esborrar l'ona */
int rms[N_MOSTRES+2];        /* Taula on es guarden els valors de l'FFT que s'han de dibuixar */
int rmsb[N_MOSTRES+2];       /* Còpia de la taula anterior que servirà per esborrar l'FFT */
int magnif=1;                /* Escala inicial de magnificació horitzontal de l'oscil.loscopi */
int q=10;                    /* Nombre de quadrats de la representació temporal */
int escfft=5;                /* Factor inicial d'escalat vertical de l'FFT */
int simul=3;                 /* Tipus d'ona amb la qual es simula una entrada */
                             /* simul=0, entrada normal; simul=(1..4), entrada simulada */


void ini_graf(void)
        /*******************************************************/
        /*****                   ini_graf()              *****/
        /*****   Procediment que inicialitza la pantalla gràfica en   *****/
        /*****   mode VGA.                               *****/
        /*******************************************************/
{
 int GraphDriver,GraphMode,ErrorCode;
 GraphDriver = DETECT;                 /* Demanada l'auto-detecció */
 initgraph(&GraphDriver,&GraphMode, "");
 ErrorCode = graphresult();            /* Lectura del resultat de la inicialització */
 if (ErrorCode != grOk)                /* Ha ocurregut un error durant la inicialització */
 {
  printf(" ERROR DE GRAFICS DEL SISTEMA: %s\n", grapherrormsg(ErrorCode) );
  exit(1);
 }
}
```

```
void simular(void)
        /**************************************************************/
        /*****                    simular()                       *****/
        /*****   Procediment que simula una ona d'entrada, omplint *****/
        /*****   una taula de "N_MOSTRES", amb valors compresos    *****/
        /*****   entre 0 i 159 (0 -> Vi=-5V;  159 -> Vi=+5V).      *****/
        /**************************************************************/
{
 int n,t=0;
 for (n=0;n<N_MOSTRES;n++) {
                /***** Ona sinusoïdal de freqüència f *****/
    if (simul==1) {
        taula[n]=80+(float)(sin(n*6.2831853/10))*80;
        if (taula[n]>159) taula[n]=159;
        if (taula[n]<0) taula[n]=0;
    }
                /***** Suma de dues ones sinusoïdals de freq. diferent *****/
    if (simul==2) {
        taula[n]=80+((float)(0.75*sin(n*6.2831853/70))+(float)0.25*sin(n*6.2831853/30))*80;
        if (taula[n]>159) taula[n]=159;
        if (taula[n]<0) taula[n]=0;
    }
                /***** Suma de tres ones sinusoïdals de freq. diferent *****/
    if (simul==3) {
        taula[n]=80+(0.6*(float)(sin(n*6.2831853/40))+0.4*(float)sin(n*6.2831853/20)+
        +0.2*(float)sin(n*6.2831853/5))*80;
        if (taula[n]>159) taula[n]=159;
        if (taula[n]<0) taula[n]=0;
    }
                /***** Ona quadrada de freqüència f *****/
    if (simul==4) {
        taula[n]=0;
        t++;
        if (t>10) taula[n]=159;
        if (t>20) t=0;
        if (taula[n]>159) taula[n]=159;
        if (taula[n]<0) taula[n]=0;
    }
  }
}


void dib_oscil(void)
        /*******************************************************/
        /*****                 dib_oscil()                 *****/
        /*****  Procediment que dibuixa els 512 valors de la taula *****/
        /*****  "taula[n]" en la pantalla de l'oscil.loscopi, i    *****/
        /*****  també dibuixa la graella.                    *****/
        /*******************************************************/
{
 int n,k,x,y,m,xant,yant,xantb,yantb;
 int qq=51.2*q;                    /* Longitud en punts que tindrà la representació */
 setviewport(x1,y1,x1+q*51.2,y1+159,1);/* Finestra activa=pantalla oscil.loscopi */
```

```
 xant=0;                                /* Valor inicial "X" per dibuixar */
 xantb=0;                               /* Valor inicial "X" per esborrar */
 yant=taula[0];                         /* Valor inicial "Y" per dibuixar */
 yantb=taulab[0];                       /* Valor inicial "Y" per esborrar */
 setlinestyle(0,0,1);
 setcolor(0);
 moveto(xantb,yantb);                   /* Es posiciona al punt inicial */
 taulab[0]=taula[0];                    /* Posa el 1r valor a la taula d'esborrar */
 k=0;
 for (n=0;n<qq;n=n+magnif)              /* Per tots els punts que s'han de dibuixar... */
 {
  setcolor(0);                          /* Esborra el primer segment */
  y=taulab[k];
  line(n,y,xantb,yantb);
  xantb=n;                              /* El punt inicial del següent segment... */
  yantb=y;                              /* ... és el punt final del segment anterior */
  setcolor(10);                         /* Dibuixa el primer segment */
  y=taula[k];
  line(n,y,xant,yant);
  xant=n;                               /* El punt inicial del següent segment... */
  yant=y;                               /* ... és el punt final del segment anterior */
  taulab[k]=y;
  k++;                                  /* Passa al següent segment */
 }
       /***** dibuix de la graella *****/
 setviewport(0,0,639,479,1);
 setcolor(3);
 setlinestyle(1,0,1);
 for (n=0;n<=4;n++) line(x1,y1+n*40,x1+q*51.2,y1+n*40);
 for (n=0;n<=q;n++) line(x1+n*51.2,y1,x1+n*51.2,y1+160);
}


int bitrev(int j, int nu)
       /**************************************************/
       /*****                 bitrev()              *****/
       /*****   Rutina "Bit Reverse" que s'utilitza en el càlcul de  *****/
       /*****   l'FFT.                                *****/
       /**************************************************/
{
 int i,j1,j2,k;
 j1=j;
 k=0;
 for (i=1;i<=nu;i++)
 {
  j2=j1/2;
  k=k*2+(j1-2*j2);
  j1=j2;
 }
 return(k);
}
```

```
void fft(void)
        /********************************************************/
        /*****                   fft()                     *****/
        /*****   Rutina que calcula l'FFT d'una taula "taula[k]" de   *****/
        /*****   "N_MOSTRES" mostres. La rutina retorna l'FFT amb    *****/
        /*****   freqüències positives en la taula "rms[n]", també de *****/
        /*****   "N_MOSTRES" valors, interpolant els valors senars.  *****/
        /********************************************************/
{
 int nu,n2,nu1,i,l,k;
 float treal,timag,p,arg,c,s,m;
 float ximag[N_MOSTRES+1],xreal[N_MOSTRES+1];
 char ch;
 nu=(log10(N_MOSTRES)/log10(2));
 n2=N_MOSTRES/2;
 m=6.283185/N_MOSTRES;
 nu1=nu-1;
 for (k=0;k<=N_MOSTRES;k++)
 {
  ximag[k]=0;                       /* L'FFT és complex, i aquí només s'utilitza */
  xreal[k]=(float)taula[k];         /* el valor real. */
 }
 k=0;
 for (l=1;l<=nu;l++)
 {
  lbl:for (i=1;i<=n2;i++)
  {
   p=bitrev(k/pow(2,nu1),nu);
   arg=m*p;
   c=cos(arg);
   s=sin(arg);
   treal=xreal[k+n2]*c+ximag[k+n2]*s;
   timag=ximag[k+n2]*c-xreal[k+n2]*s;
   xreal[k+n2]=xreal[k]-treal;
   ximag[k+n2]=ximag[k]-timag;
   xreal[k]=xreal[k]+treal;
   ximag[k]=ximag[k]+timag;
   k++;
  }
  k=k+n2;
  if (k<N_MOSTRES) goto lbl;
  k=0;
  nu1--;
  n2=n2/2;
 }
 for (k=1;k<N_MOSTRES-1;k++)
 {
  i=bitrev(k,nu);
  if (i>k)
  {
   treal=xreal[k];
   timag=ximag[k];
   xreal[k]=xreal[i];
   ximag[k]=ximag[i];
   xreal[i]=treal;
   ximag[i]=timag;
  }
```

```
 }
 rms[0]=0;                                  /* S'omple la taula "rms[n]" amb el valor final de l'FFT */
 for (k=1;k<N_MOSTRES/2;k++)
 {
  rms[(k*2)]=escfft*0.8*(sqrt(pow(xreal[k]/(float)N_MOSTRES,2)+pow(ximag[k]/(float)N_MOSTRES,2)));
  rms[(k*2)-1]=(rms[k*2]+rms[(k*2)-2])/2;
 }
}



void dib_FFT(void)
      /*************************************************/
      /*****              dib_FFT()               *****/
      /*****  Procediment que dibuixa els 512 valors de la taula  *****/
      /*****  "rms[n]" en la pantalla de l'FFT, i també dibuixa  *****/
      /*****  la graella.                         *****/
      /*************************************************/
{
 int n,x,y,top=160;
 double k;
 int xant=0,xantb=0,yant=160,yantb=y2+160;
 setviewport(x2,y2,x2+512,y2+160,1);
 setlinestyle(0,0,1);
 setcolor(14);
 moveto(xant,yant);
 rmsb[0]=rms[0];
 setlinestyle(0,0,1);
 for (n=1;n<512;n++)
 {
   setcolor(0);
   y=top-1-(int)rmsb[n];
   line(n,y,xantb,yantb);
   xantb=n;
   yantb=y;
   setcolor(14);
   y=top-1-(int)rms[n];
   line(n,y,xant,yant);
   xant=n;
   yant=y;
   rmsb[n]=rms[n];
 }
      /***** Dibuix de la graella *****/
 setviewport(0,0,639,479,1);
 setcolor(3);
 setlinestyle(1,0,1);
 for (n=0;n<=4;n++) line(x2,y2+n*40,x2+512,y2+n*40);
 for (n=0;n<=9;n++) line(x2+n*51.2,y2,x2+n*51.2,y2+160);
 line(x2+512,y2,x2+512,y2+160);
 setcolor(11);
 settextstyle(SMALL_FONT,HORIZ_DIR,5);
 outtextxy(x2-38,y2,"100%");
 outtextxy(x2-31,y2+37,"75%");
 outtextxy(x2-31,y2+77,"50%");
 outtextxy(x2-31,y2+117,"25%");
 outtextxy(x2-24,y2+153,"0%");
}
```

```
void teclat(void)
        /*****************************************************/
        /*****                   teclat()               *****/
        /*****   Procediment que gestiona i controla les tecles  *****/
        /*****   premudes del teclat.                   *****/
        /*****************************************************/
{
 int c;
 if (kbhit())                          /* Si s'ha premut una tecla..... */
 {
  c=getche();
  switch(c)
  {
   case 27: exit(0);                   /* Si la tecla és "Esc", sortida del programa */
            break;
  }
 }
}



main()
        /*****************************************************/
        /*****                   main()                 *****/
        /*****        Procediment principal del programa     *****/
        /*****************************************************/
{
 ini_graf();                           /* Inicialitza els gràfics */
 setpalette(10,46);                    /* Color del traç de l'oscil.loscopi */
 setpalette(14,54);                    /* Color del traç de l'FFT */
 simular();                            /* Simula una ona d'entrada */
 do                                    /* Repeteix continuament..... */
 {
  dib_oscil();                            /* Dibuixa el senyal */
  fft();                                  /* Calcula l'FFT */
  dib_FFT();                              /* Dibuixa l'FFT */
  teclat();                               /* Gestiona les tecles premudes */
 } while (1);                          /* Repeteix continuament..... */
}
```

```
/*
   ***************************************************************
   * Program      : ADBLOCK.C                                    *
   * Description : Demo program for PCL-711 pacer trigger A/D    *
   *               conversion on a block of channels with interrupt *
   *               data transfer.                                *
   * Revision     : 1.00                                         *
   * Date         : 6/03/92              Advantech Co., Ltd.     *
   ***************************************************************
*/

#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <dos.h>

extern pcl711(int, unsigned int *);
unsigned int param[60];                 /* If two boards installed, need to declare
                                           the second parameter array */
unsigned int data[100];                 /* Conversion data buffer */
unsigned int far * dat;
main()
{
unsigned int i;
float       DataBuf;

    dat = data;
    clrscr();
    param[0]  = 0;                      /* Board number                          */
    param[1]  = 0x220;                  /* Base I/O address                      */
    param[4]  = 2;                      /* IRQ level    : IRQ2                   */
    param[5]  = 50;                     /* Pacer rate = 2M / (50 * 100) = 400 Hz */
    param[6]  = 100;
    param[7]  = 0;                      /* Trigger mode, 0 : pacer trigger       */
    param[8]  = 0;                      /* Non-cyclic                            */
    param[10] = FP_OFF(dat);            /* Offset of A/D data buffer A           */
    param[11] = FP_SEG(dat);            /* Segment of A/D data buffer A          */
    param[12] = 0;                      /* Data buffer B address, if not used,   */
    param[13] = 0;                      /* must set to 0.                        */
    param[14] = 50;                     /* A/D conversion number                 */
    param[15] = 0;                      /* A/D conversion start channel          */
    param[16] = 7;                      /* A/D conversion stop channel           */
    param[17] = 0;                      /* Overall gain code, 0 : +/- 5V         */

    /* param[18] = FP_OFF(gain_array);
       param[19] = FP_SEG(gain_array);  */

    /* param[45] : Error code
       param[46] : Return value 0
       param[47] : Return value 1 */

    pcl711(3, param);                   /* Func 3 : Hardware initialization */
    if (param[45] != 0) {
        printf("\nDRIVER INITIALIZATION FAILED !");
        exit(1);
        }
```

```
pcl711(100, param);                    /* Func 100 : A/D initialization */
if (param[45] != 0) {
    printf("\nA/D INITIALIZATION FAILED !");
    exit(1);
    }

/* Block channel scan function 105 is very similar to function 9.
   The only difference is block channel scan function 105 scans
   all input channels (specified by param[15] and param[16])
   on each trigger pulse, and the function 9 scans one channel
   on each trigger pulse. */

pcl711(105, param);                    /* Func 105 : Pacer trigger A/D conversion */
if (param[45] != 0) {                  /* with interrupt data transfer */
    printf("\n A/D INTERRUPT BLOCK DATA TRANSFER FAILED !");
        exit(1);
        }
do {
        pcl711(106, param);            /* Func 106:Check interrupt status */
}while((param[46] & 1) != 0);          /* 0 : not active, 1 : active */

for (i = 0; i < param[14]; i++)        /* Display data */
    {
      DataBuf = data[i] & 0xFFF;
      DataBuf =( (5 - (-5)) * DataBuf / 4096) + (-5);
      /*
        (5 - (-5)) : A/D input range (-5V  to  5V)
        4096       : Full scale 12 bit A/D data
        DataBuf    : A/D input data
        (-5)       : A/D input range "-5" V
      */
      printf("\ndata[%3d] = % 1.2f V ", i, DataBuf);

    }
}
```